What I won't talk about

Luc De Raedt (KULeuven)

Dagstuhl Seminar on ML and Formal Methods

September 2017

TaCLe

Learning constraints in spreadsheets and tabular data

Sergey Paramonov, Samuel Kolb, Tias Guns, Luc De Raedt

KU Leuven

(Machine Learning 2017, ECMLPKDD Track, CIKM 2017 demo track)

Reverse Engineering Formulae / Constraints

Illustration

ID	Salesperson	1st Quarter	2nd Quarter	3rd Quarter	4th Quarter	Total	Rank	Label	Items sold total	Max items sold
1	Diana Coolen	353	378	396	387	1514	2	Great	34	20
2	Marc Desmet	370	408	387	386	1551	1	Great	29	10
3	Kris Goossens	175	146	167	203	691	3	Low	19	19
4	Birgit Kenis	93	98	96	105	392	4	Low	17	15

T1

B1 = T1[:, 1] B2 = T1[:, 2] B3 = T1[:, 3:8] B4 = T1[:, 9] B5 = T1[:, 10:11]

T2

Total	991	1030	1046	1081	4148
Average	247.75	257.5	261.5	270.25	1037
Max	370	408	396	387	1551
Min	93	98	96	105	392

B6 = T2[1:4, :]

Salesperson	Items sold
Diana Coolen	5
Marc Desmet	10
Marc Desmet	8
Diana Coolen	9
Birgit Kenis	15
Marc Desmet	8
Birgit Kenis	2
Diana Coolen	20
Marc Desmet	3
Kris Goossens	19

T3

B8 = T3[:, 1] B9 = T3[:, 2]

Quarter	Income	Expenses	Total
Q1	991	212	779
Q2	1030	710	1099
Q3	1046	137	2008
Q4	1081	240	2849

T4

B10 = T4[:, 1] B11 = T4[:, 2:4]

Customer	Contact	Contact Name
Frank	1	Diana Coolen
Sarah	3	Kris Goossens
George	3	Kris Goossens
Mary	2	Diana Coolen
Tim	4	Birgit Kenis

T5

B12 = T5[:, 1] B13 = T5[:, 2] B14 = T5[:, 3]

$\mathit{SERIES}(T_1[:,1])$

$T_1[:,1] = \mathit{RANK}(T_1[:,5])^*$

$T_1[:,1] = \mathit{RANK}(T_1[:,6])^*$

$T_1[:,1] = \mathit{RANK}(T_1[:,10])^*$

$T_1[:,8] = \mathit{RANK}(T_1[:,7])$

$T_1[:,8] = \mathit{RANK}(T_1[:,3])^*$

$T_1[:,8] = \mathit{RANK}(T_1[:,4])^*$

$T_1[:,7] = \mathit{SUM}_{row}(T_1[:,3{:}6])$

$T_1[:,10] = \mathit{SUMIF}(T_3[:,1], T_1[:,2], T_3[:,2])$

$T_2[1,:] = \mathit{SUM}_{col}(T_1[:,3{:}7])$

$T_2[2,:] = \mathit{AVERAGE}_{col}(T_1[:,3{:}7])$

$T_2[3,:] = \mathit{MAX}_{col}(T_1[:,3{:}7])$

$T_2[4,:] = \mathit{MIN}_{col}(T_1[:,3{:}7])$

$T_4[:,2] = \mathit{SUM}_{col}(T_1[:,3{:}6])$

$T_4[:,4] = PREV(T_4[:,4]) + T_4[:,2] - T_4[:,3]$

$T_5[:,2] = \mathit{LOOKUP}(T_5[:,3], T_1[:,2], T_1[:,1])^*$

$T_5[:,3] = \mathit{LOOKUP}(T_5[:,2], T_1[:,1], T_1[:,2])$

$T_1[:,11] = \mathit{MAXIF}(T_3[:,1], T_1[:,2], T_3[:,2])$

(b) Constraints learned for the tables above, except 19 *ALLDIFFERENT*, 2 *PERMUTATION* and 5 *FOREIGNKEY* and 5 *ASCENDING* constraints not shown. Constraints marked with $*$ were not present in the original spreadsheets.

We are working on learning constraints and CSPs

What I will talk about

Luc De Raedt (KULeuven)

Dagstuhl Seminar on ML and Formal Methods

August 2017

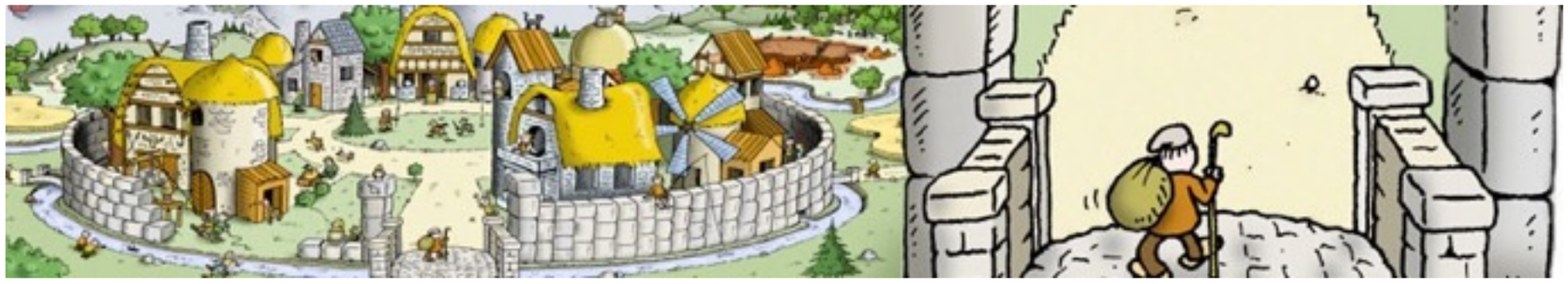
Dynamic Probabilistic Logic Programs

Luc De Raedt (KULeuven)

Dagstuhl Seminar on ML and Formal Methods

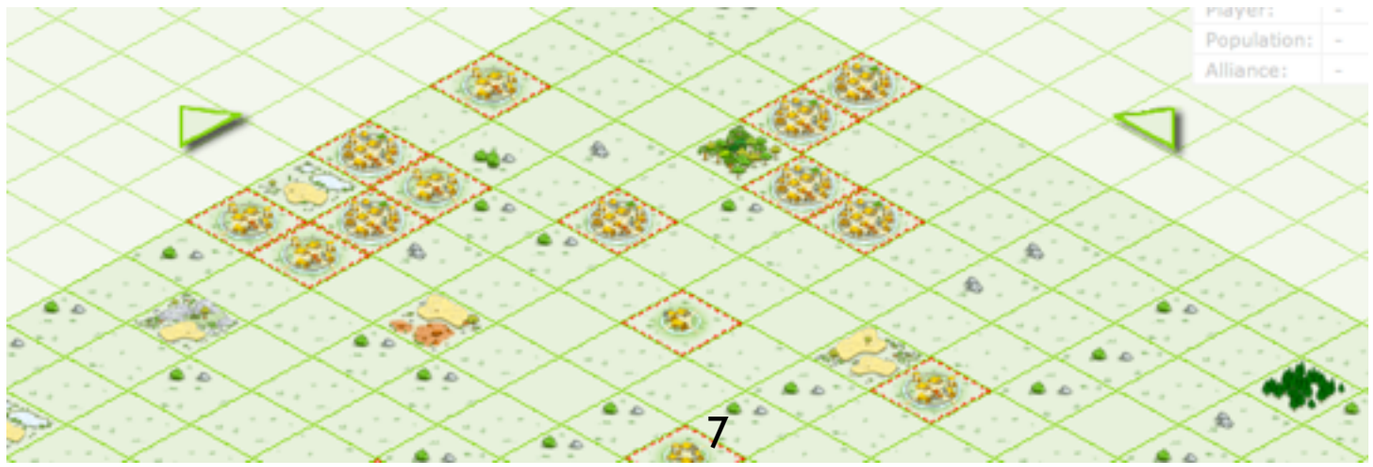
August 2017

Dynamics: Evolving Networks



- *Travian*: A massively multiplayer real-time strategy game
- Commercial game run by TravianGames GmbH
- ~3.000.000 players spread over different "worlds"
- ~25.000 players in one world

[Thon et al. MLJ 11]



World Dynamics

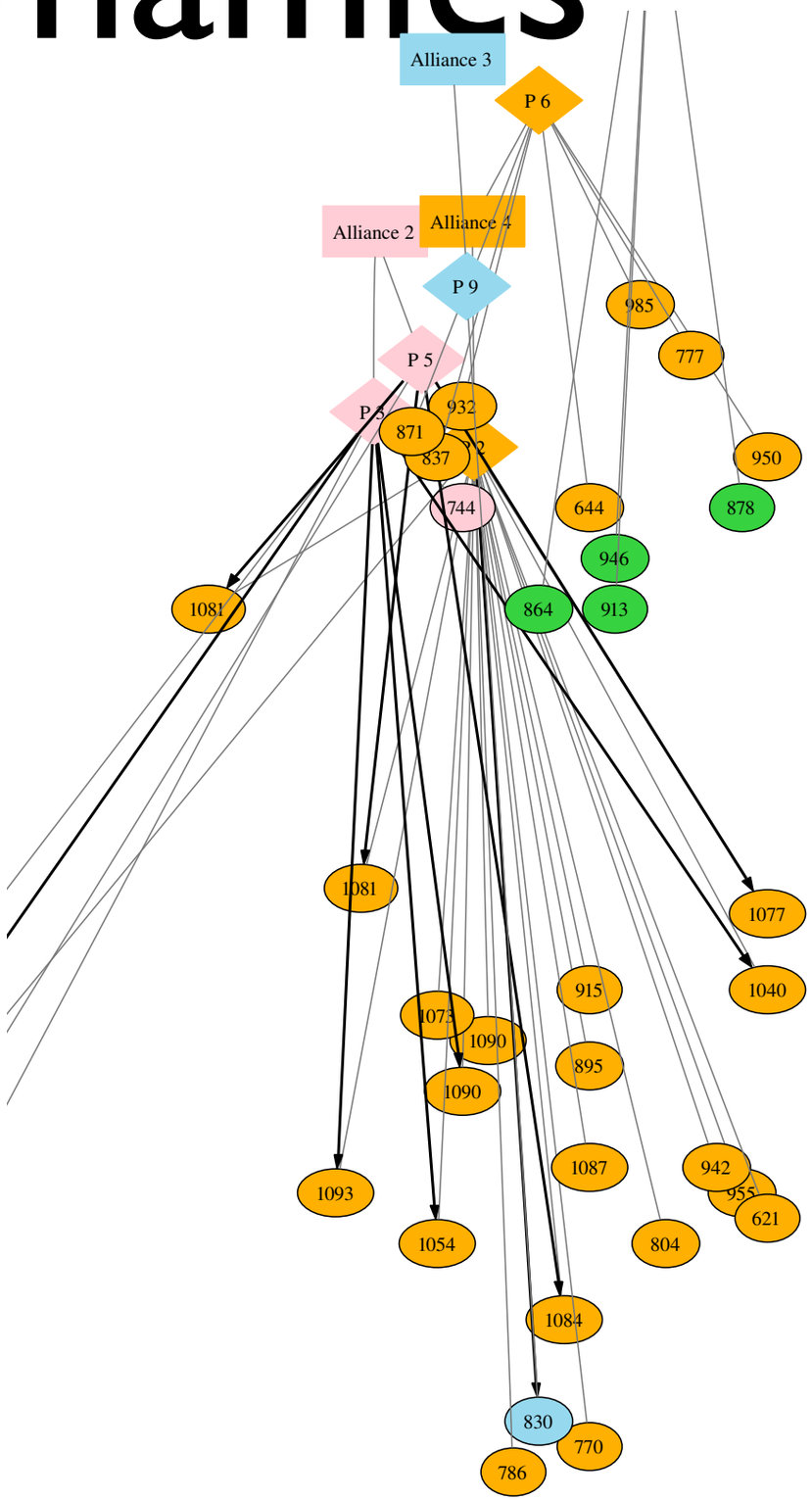
Fragment of world with

~10 alliances
~200 players
~600 cities

alliances color-coded

Can we build a model
of this world ?
Can we use it for playing
better ?

[Thon, Landwehr, De Raedt, ECML08]



World Dynamics

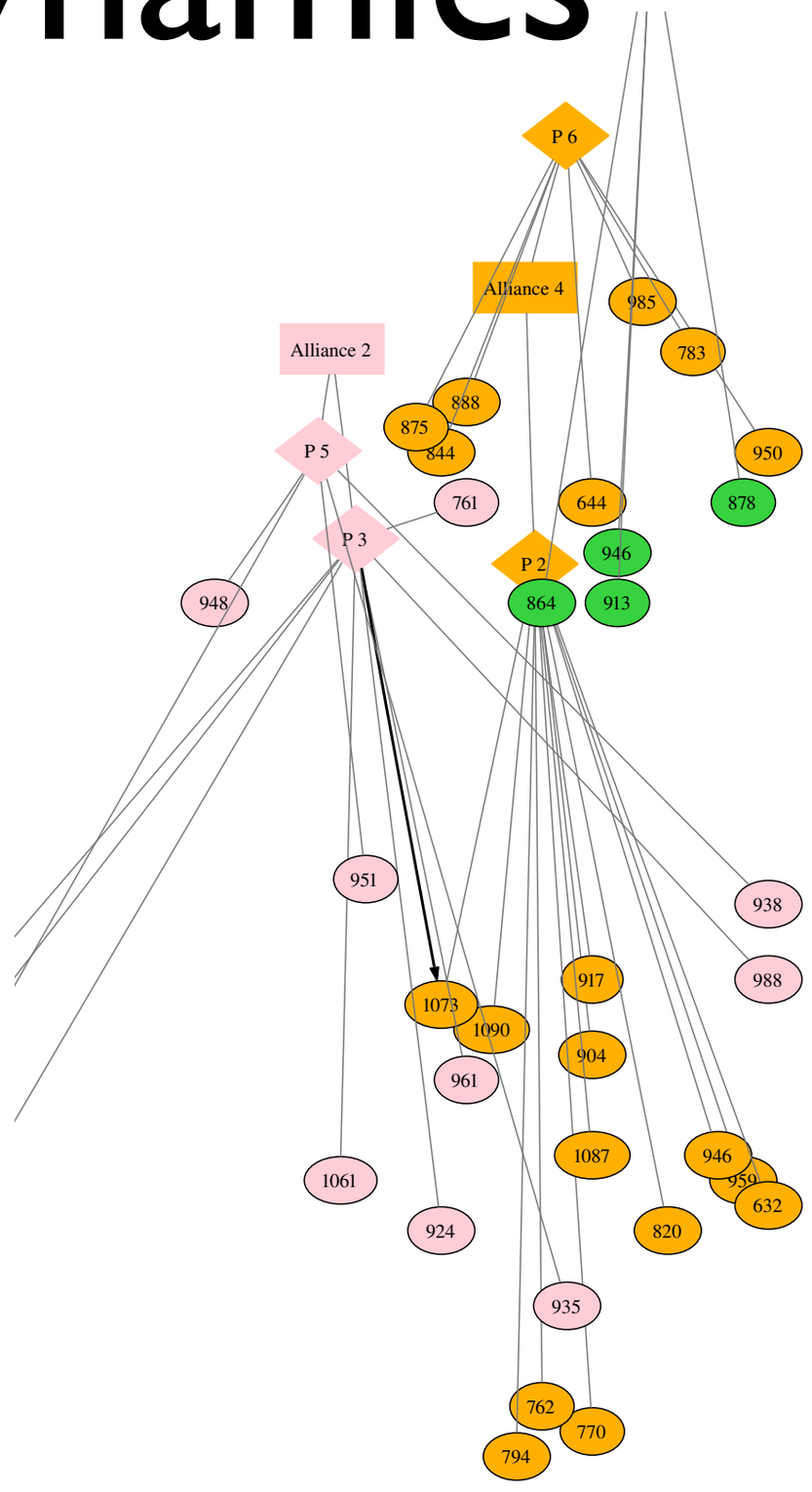
Fragment of world with

~10 alliances
~200 players
~600 cities

alliances color-coded

Can we build a model
of this world ?
Can we use it for playing
better ?

[Thon, Landwehr, De Raedt, ECML08]



World Dynamics

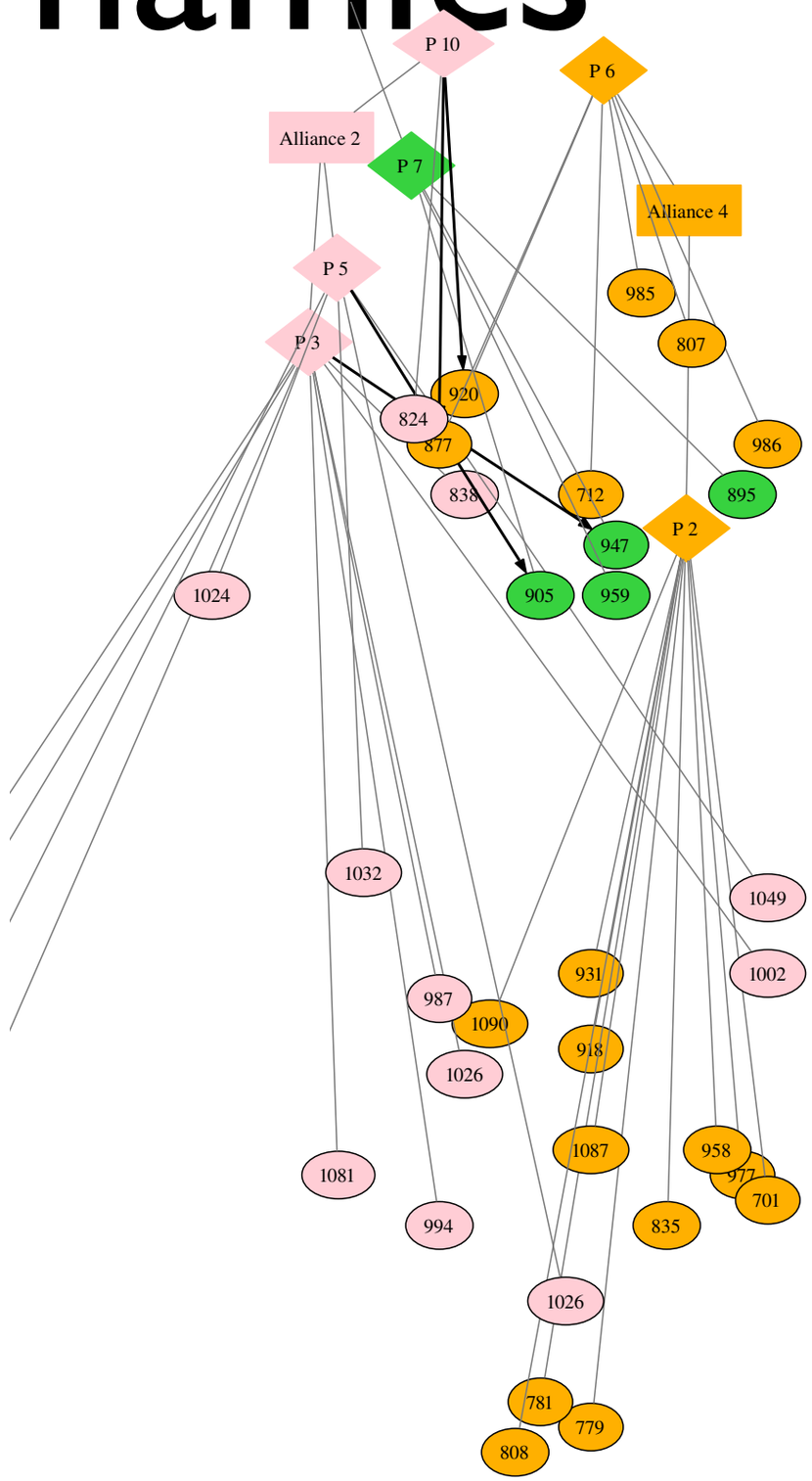
Fragment of world with

~10 alliances
~200 players
~600 cities

alliances color-coded

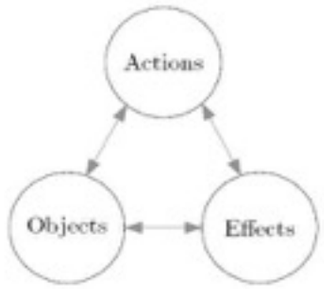
Can we build a model
of this world ?
Can we use it for playing
better ?

[Thon, Landwehr, De Raedt, ECML08]



Learning relational affordances

Learn probabilistic model

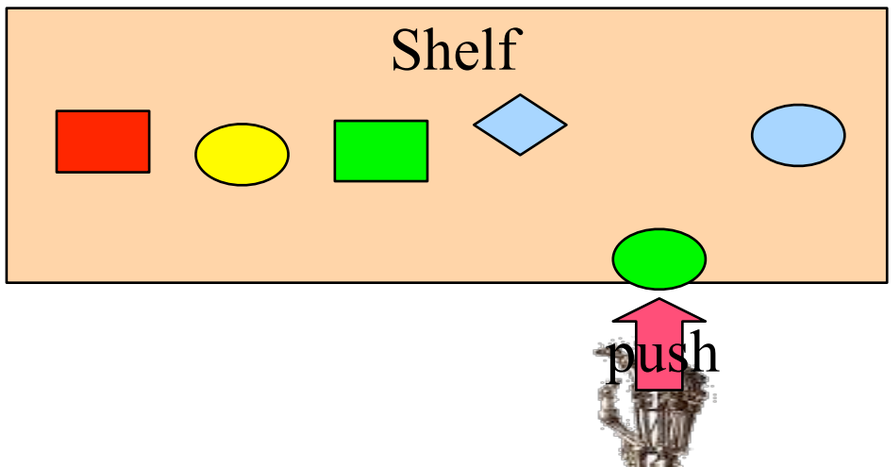


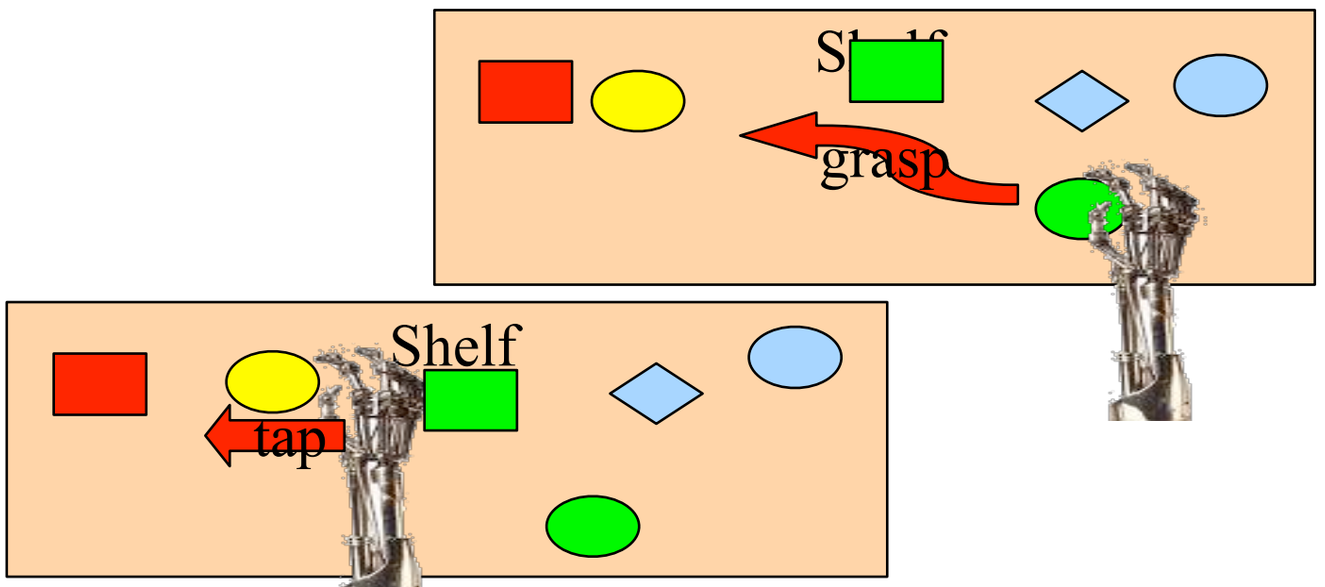
Inputs	Outputs	Function
(O, A)	E	Effect prediction
(O, E)	A	Action recognition/planning
(A, E)	O	Object recognition/selection

Learning relational affordances between two objects (learnt by experience)

Moldovan et al. ICRA 12, 13, 14, PhD 15
Nitti et al MLJ 15, 17 (forthcoming), Phd 16

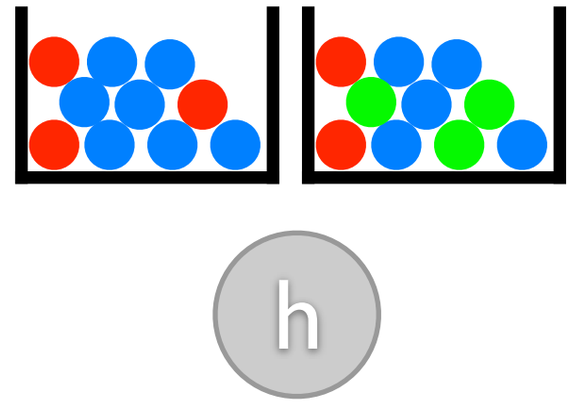
From two object interactions
Generalize to N





ProbLog by example:

A bit of gambling



- toss (biased) coin & draw ball from each urn
- win if (heads and a red ball) or (two balls of same color)

```
0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue) <- true.
0.2 :: col(2,red); 0.3 :: col(2,green);
                   0.5 :: col(2,blue) <- true.

win :- heads, col(_,red).
win :- col(1,C), col(2,C).
```

probabilistic fact: heads is true with probability 0.4 and false with 0.6)

annotated disjunction: first ball is red with probability 0.3 and blue with 0.7

annotated disjunction: second ball is red with probability 0.2, green with 0.3, and blue with 0.5

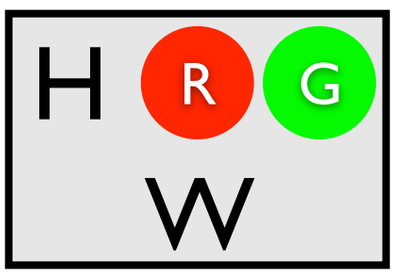
logical rule encoding background knowledge

consequences

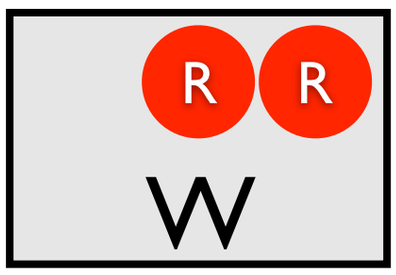
Possible Worlds

```
0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue) <- true.
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue) <- true.

win :- heads, col( ,red).
win :- col(1,C), col(2,C).
```

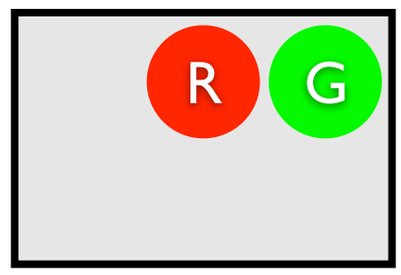
$0.4 \times 0.3 \times 0.3$



$(1-0.4) \times 0.3 \times 0.2$



$(1-0.4) \times 0.3 \times 0.3$



Questions

```
0.4 :: heads.

0.3 :: col(1,red); 0.7 :: col(1,blue) <- true.
0.2 :: col(2,red); 0.3 :: col(2,green); 0.5 :: col(2,blue) <- true.

win :- heads, col(_,red).
win :- col(1,C), col(2,C).
```

marginal probability

- Probability of `win`?

conditional probability

- Probability of `win` given `col(2,green)`?

- Most probable world where `win` is true?

MPE inference

Inference is #P-complete — weighted model counting

Distributional Clauses (DC)

- Discrete- and continuous-valued random variables

random variable with Gaussian distribution

```
length(Obj) ~ gaussian(6.0,0.45) :- type(Obj,glass).
stackable(OBot,OTop) :-
    ≃length(OBot) ≥ ≃length(OTop),
    ≃width(OBot) ≥ ≃width(OTop).
ontype(Obj,plate) ~ finite([0 : glass, 0.0024 : cup,
                            0 : pitcher, 0.8676 : plate,
                            0.0284 : bowl, 0 : serving,
                            0.1016 : none])
                    :- obj(Obj), on(Obj,O2), type(O2,plate).
```

comparing values of random variables

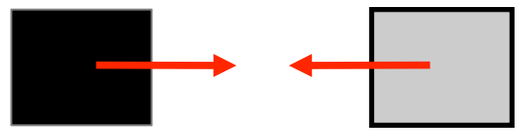
random variable with discrete distribution

[Gutmann et al, TPLP 11; Nitti et al, IROS 13]

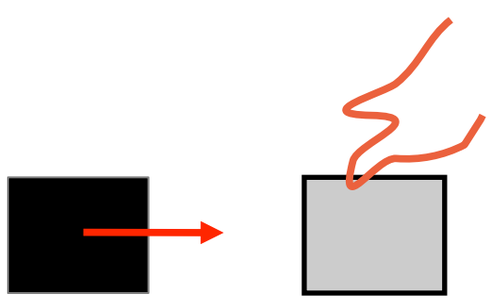
Magnetic scenario

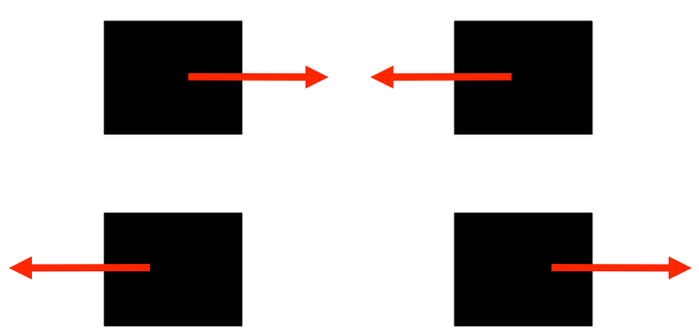
- 3 object types: magnetic, ferromagnetic, nonmagnetic

- Nonmagnetic objects do not interact
- A magnet and a ferromagnetic object attract each other



- Magnetic force that depends on the distance
- If an object is held magnetic force is compensated.





Magnetic scenario

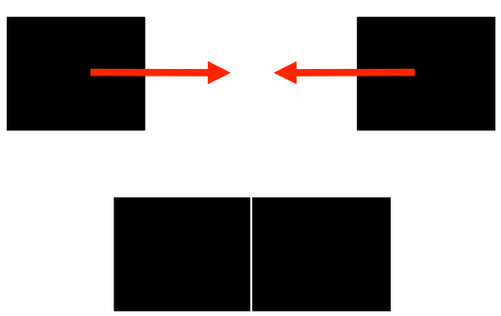
- 3 object types: magnetic, ferromagnetic, nonmagnetic

$$\text{type}(X)_t \sim \text{finite}([1/3\text{:magnet}, 1/3\text{:ferromagnetic}, 1/3\text{:nonmagnetic}]) \leftarrow \text{object}(X).$$

- 2 magnets attract or repulse

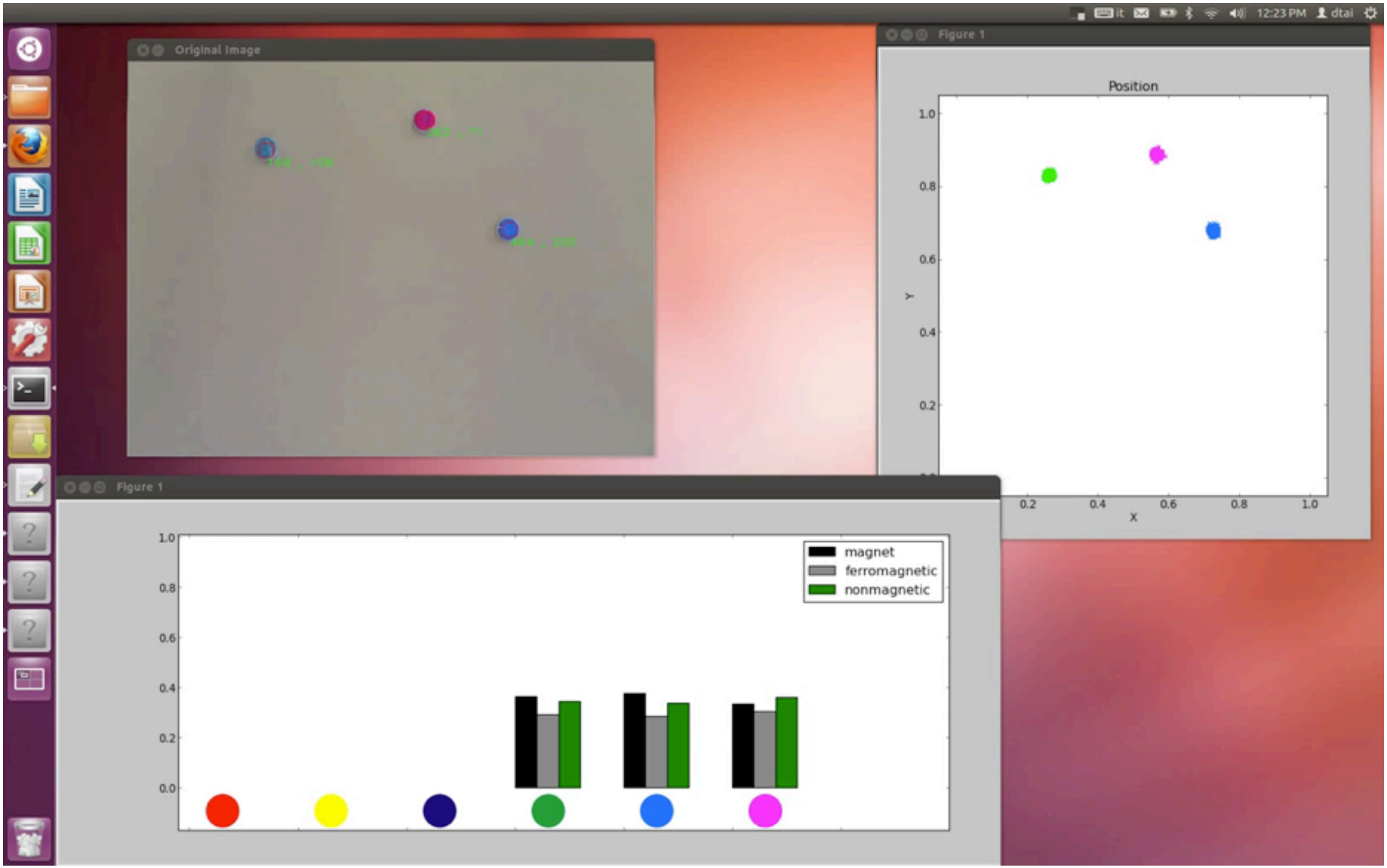
$$\text{interaction}(A,B)_t \sim \text{finite}([0.5\text{:attraction}, 0.5\text{:repulsion}]) \leftarrow \text{object}(A), \text{object}(B), A<B, \text{type}(A)_t = \text{magnet}, \text{type}(B)_t = \text{magnet}.$$

- Next position after attraction



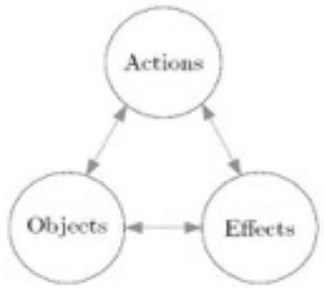
$$\text{pos}(A)_{t+1} \sim \text{gaussian}(\text{middlepoint}(A,B)_t, \text{Cov}) \leftarrow \text{near}(A,B)_t, \text{not}(\text{held}(A)), \text{not}(\text{held}(B)), \text{interaction}(A,B)_t = \text{attr}, c/\text{dist}(A,B)_t^2 > \text{friction}(A)_t.$$

$$\text{pos}(A)_{t+1} \sim \text{gaussian}(\text{pos}(A)_t, \text{Cov}) \leftarrow \text{not}(\text{attraction}(A,B)).$$


Original Image
462 , 71
160 , 108
464 , 205
Figure 1
Position
1.0
0.8
0.6
0.4
0.2
Y
0.2
0.4
0.6
0.8
1.0
X
Figure 1
1.0
0.8
0.6
0.4
0.2
0.0
magnet
ferromagnetic
nonmagnetic

Learning relational affordances

Learn probabilistic model

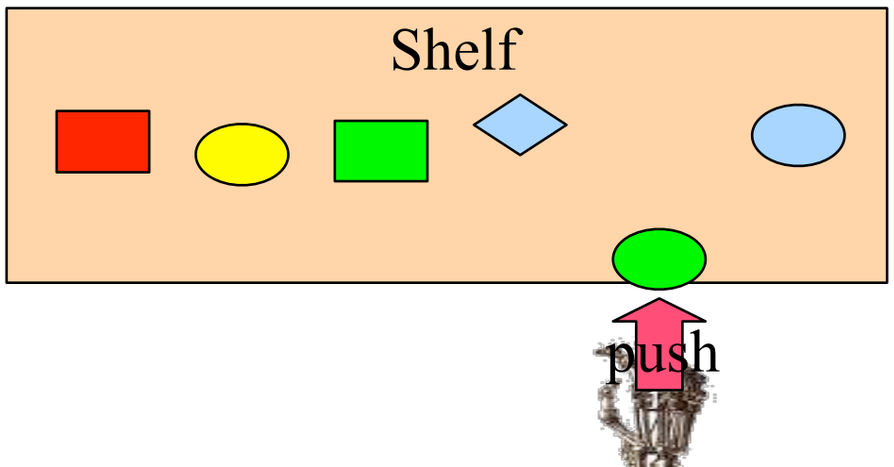


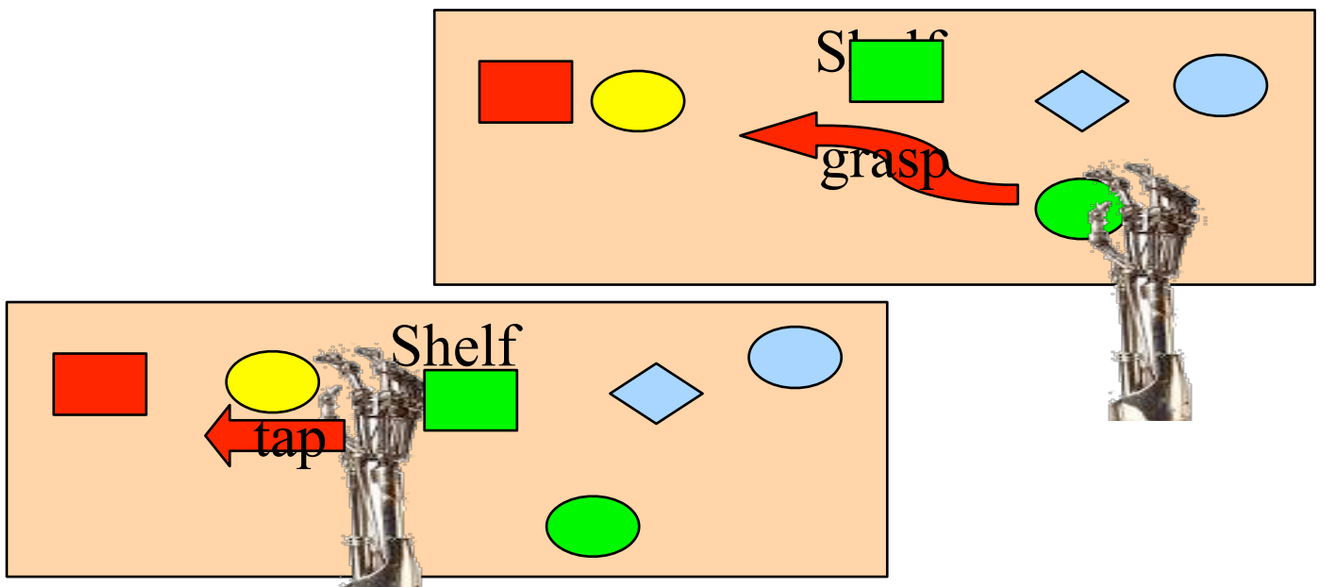
Inputs	Outputs	Function
(O, A)	E	Effect prediction
(O, E)	A	Action recognition/planning
(A, E)	O	Object recognition/selection

Learning relational affordances between two objects (learnt by experience)

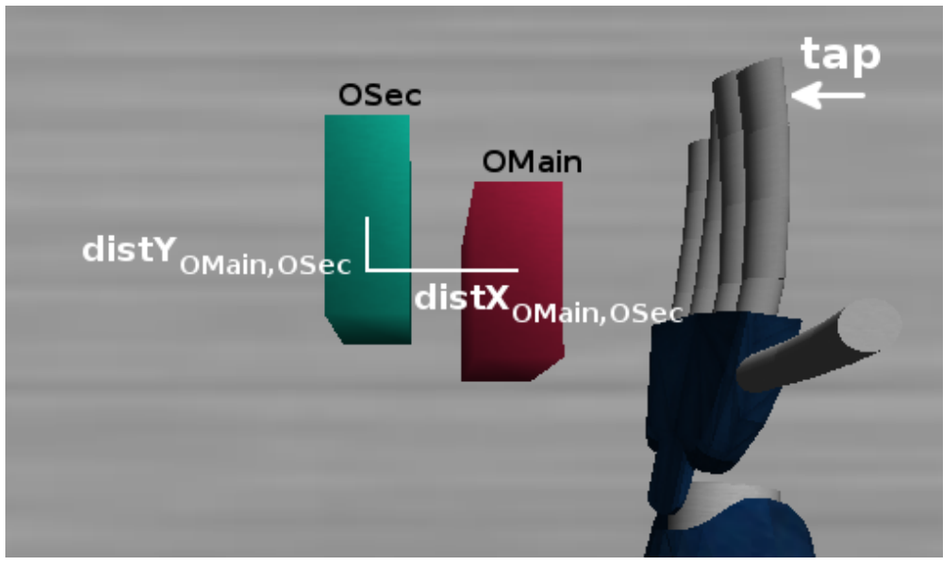
Moldovan et al. ICRA 12, 13, 14, PhD 15
Nitti et al MLJ 15, 17 (forthcoming), Phd 16

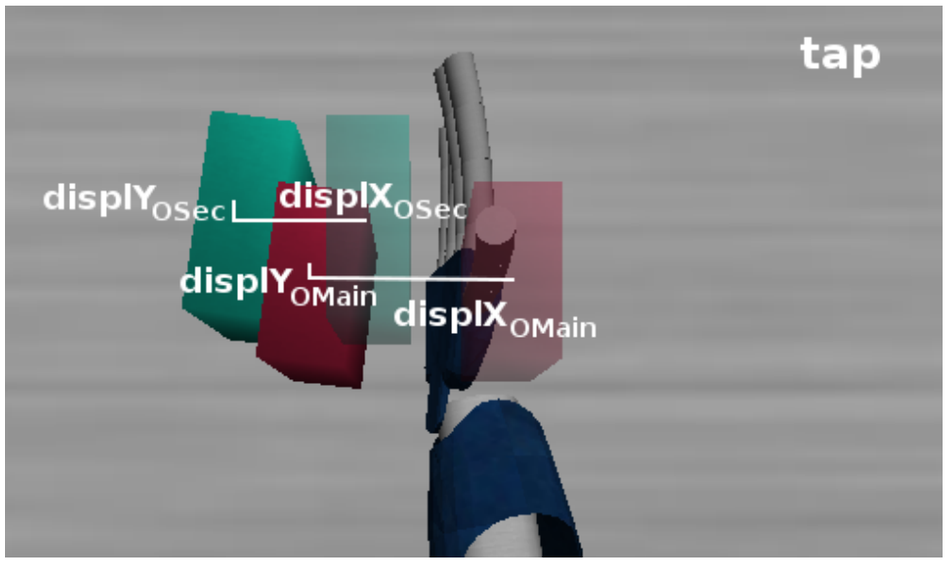
From two object interactions
Generalize to N





What is an affordance ?





Clip 8: Relational O before (l), and E after the action execution (r).

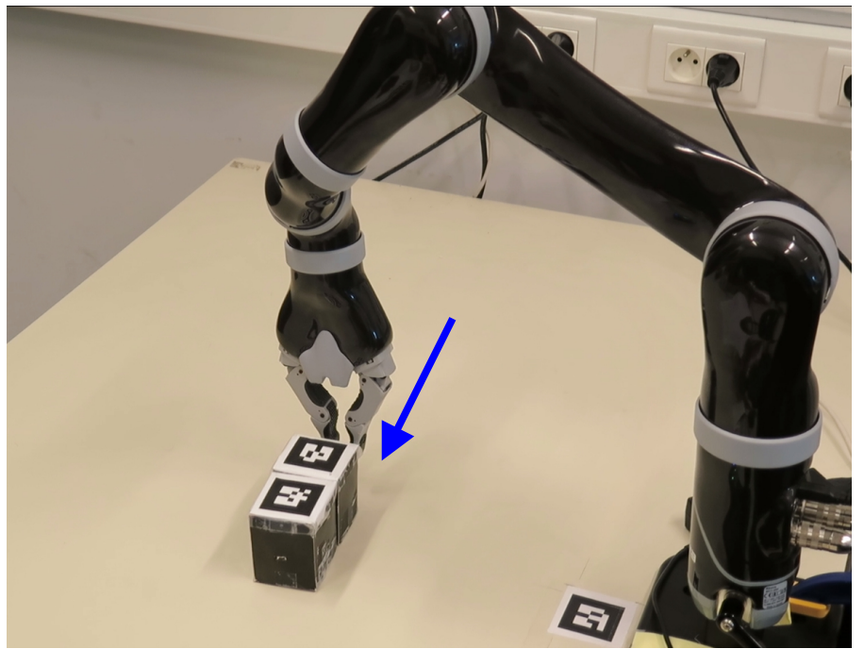
Table 1: Example collected O, A, E data for action in Figure 8

Object Properties	**Action**	**Effects**
$shape_{O_{Main}} : sprism$ $shape_{O_{Sec}} : sprism$ $distX_{O_{Main},O_{Sec}} : 6.94cm$ $distY_{O_{Main},O_{Sec}} : 1.90cm$	$tap(10)$	$displX_{O_{Main}} : 10.33cm$ $displY_{O_{Main}} : -0.68cm$ $displX_{O_{Sec}} : 7.43cm$ $displY_{O_{Sec}} : -1.31cm$

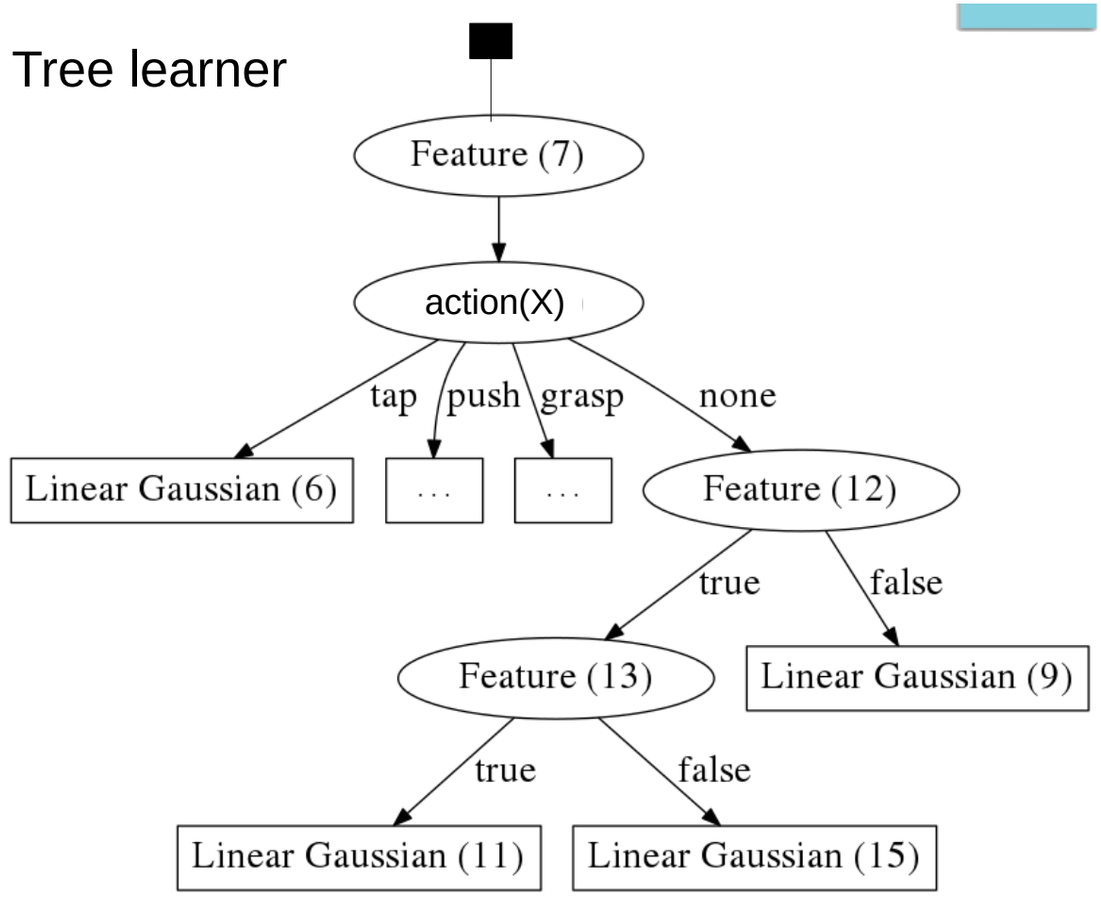
- Formalism — related to STRIPS / to PDDL but models delta
- but also joint probability model over A, E, O

Relational Affordance Learning

- **Learning the Structure of Dynamic Hybrid Relational Models**
  Nitti, Ravkic, et al. ECAI 2016
  - Captures relations/affordances
  - Suited to learn affordances in
    robotics set-up, continuous and discrete variables
  - Planning in hybrid robotics domain



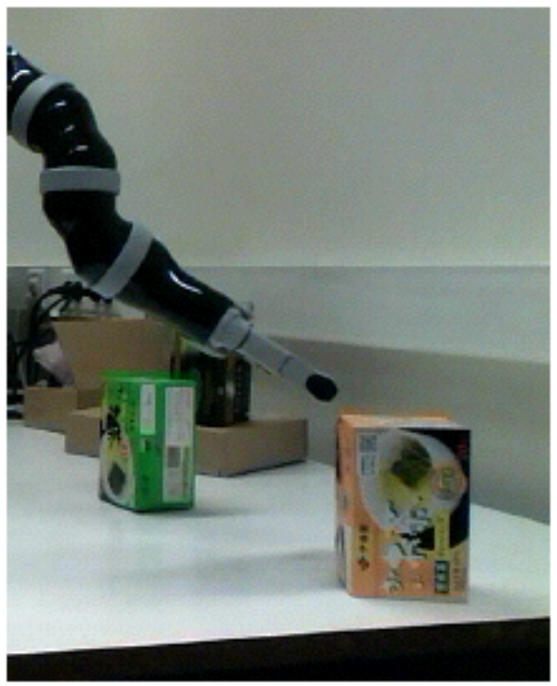
DDC Tree learner

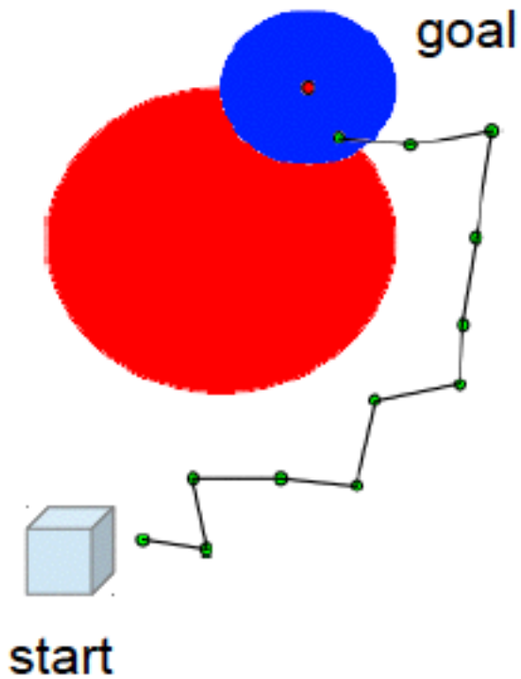


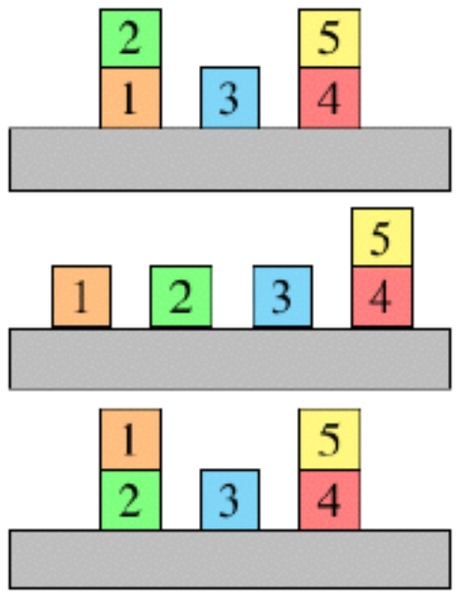
Planning

- Main task: probabilistic planning

  Find the best action to achieve the goal
- Discrete + continuous + relational representation







[Nitti et al ECML 15]

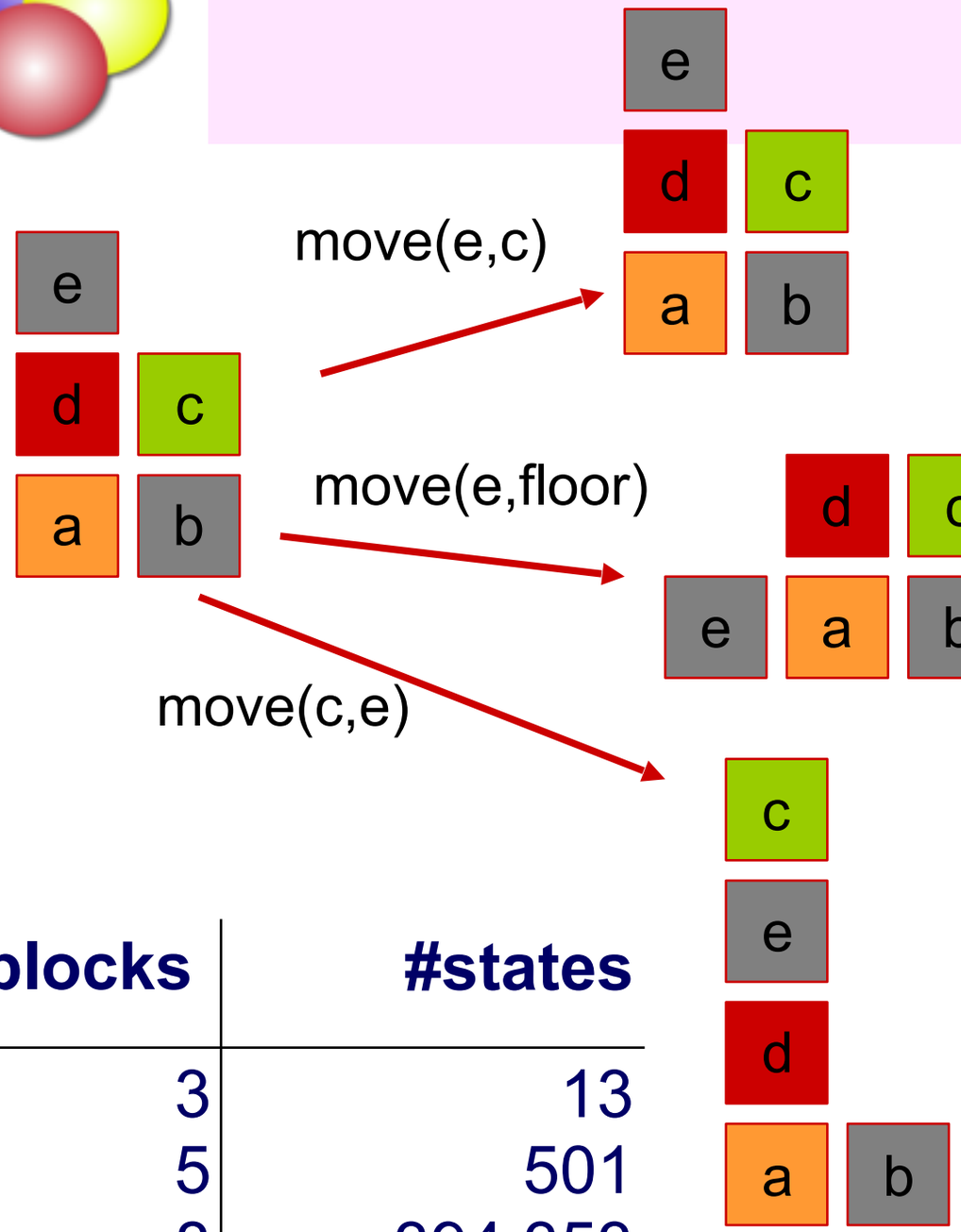
Conclusions

- Static version (~ prob. data and knowledge bases)
- Dynamic formalism is related to PDDL, can represent relational MDPs, can be learned (small problems) and can be used for planning — continuing work on scaling up
- We can learn these formalisms

Questions that I have

- What kind of relationships exist between PCTL and Prob. Planning ?
- What verification techniques work with relational worlds ? (relational MDPs versus propositional ones)
- I d like to impose constraints on what is being learned … how do I do that ?

Curse of Dimension



#blocks	#states
3	13
5	501
8	394,353
10	58,941,091

- Flat represention
- No notions of objects and relations among the objects
- Generalization (similar situations / individuals)?
- Parameter Reduction / Compression ?
  - on(a,b) for 10 blocks
    - <150 values
    - 58,941,091 states